Fondamenti di Programmazione - CdL in MATEMATICA

Appello del 6/7/2015

num. eserc.	1	2	3	4
punt. tot	7	6	7	11

N.B.:

- Negli esercizi di programmazione, viene valutata anche la leggibilità del codice proposto.
- Inoltre, non è consentito l'uso di istruzioni che alterino il normale flusso dell'esecuzione (come, ad esempio, `continue`, `break` e istruzioni di `return` all'interno di cicli che ne provochino l'uscita forzata).
- Non è consentito l'uso di variabili statiche.
- Laddove è utilizzato, il tipo `boolean` è definito da `typedef enum {false,true} boolean`.

ESERCIZIO 1 (7 punti)

- Costruire un automa non deterministico A sull'alfabeto $\Sigma = \{0, 1\}$ che accetti l'insieme delle stringhe che se interpretate in notazione binaria siano numeri divisibili per 4.
- Fornire la corrispondente grammatica **regolare**.

ESERCIZIO 2 (6 punti)
Si scriva in C una funzione **iterativa** `boolean controlla(int v[], int dim)` che presi come parametro un array `v` di interi e la sua dimensione `dim`, restituisce il valore di verità della seguente formula, dove con $\#I$ denotiamo il numero di elementi di I.

$$\#\{i \mid i \in [0, dim-1).\ (\exists j \in [i+1, dim).\ (v[i] == v[j])\} \geq 2$$

ESERCIZIO 3 (7 punti)
Si scriva in C una funzione **ricorsiva** (che non deve pertanto fare uso di costrutti iterativi, nemmeno nelle sue sottofunzioni, nel caso ce ne siano) che, dato un array `v` di interi e la sua dimensione `dim`, conta le *cime*, ovvero gli elementi che sono maggiori di tutti gli elementi successivi. Si supponga che l'ultimo elemento dell'array non possa essere una *cima*. Ad esempio, se la sequenza memorizzata fosse `3 6 2 4 3` l'output sarebbe `2`.

ESERCIZIO 4 (11 punti)
Si supponga di lavorare con liste di interi così definite:

```
struct el {int info;
           struct el *next;
           };
typedef struct el ElementoDiLista;
typedef ElementoDiLista *ListaDiInteri;
```

- Si scriva una procedura in C che, dati attraverso opportuni parametri una lista di interi e un intero $n > 0$, elimina dalla lista i primi n elementi positivi.
- Scrivere una procedura **ricorsiva** in C che, prese attraverso opportuni parametri due liste di interi ordinate in ordine strettamente crescente, aggiunge alla seconda lista tutti gli elementi della prima che non appartengono anche alla seconda, senza curarsi dell'ordine.